CHALMERS

Identifying Friction between Large-scale Agile Software Development and Plan-based Business Processes

Master of Science Thesis in the Programme Software Engineering and Technology

TOBIAS ALTEHED
ROBIN INGMARSSON

Chalmers University of Technology
University of Gothenburg
Department of Computer Science and Engineering
Göteborg, Sweden, June 2011

Identifying Friction between Large-scale Agile Software Development and Plan-based Business Processes

TOBIAS ALTEHED,
ROBIN INGMARSSON

Examiner: HELENA HOLMSTRÖM OLSSON

Chalmers University of Technology
University of Gothenburg
Department of Computer Science and Engineering
SE-412 96 Göteborg
Sweden
Telephone + 46 (0)31-772 1000

Department of Computer Science and Engineering
Göteborg, Sweden June 2011

Acknowledgements

The Authors would like to extend gratitude and thanks to Ericsson AB for providing time and resources contributing to the writing of this thesis. In particular, the authors are thankful for the support and guidance given by Mats Lindén and Heike Thiede throughout the conducted study, alongside the important contribution given by interviewees at the studied Ericsson unit. Furthermore, the Authors would like to thank Associate Professor Robert Feldt, Chalmers University of Technology, for his academic supervision and support from initial thesis proposition to thesis completion.

Preface

This Master of Science Thesis is reported here in a hybrid format, i.e. the main content of the Work is reported as a scientific article conforming to the Empirical Software Engineering Journal's template, complemented by additional appendices relevant for the examination at Chalmers University of Technology.

Table of Content

myjournal manuscript No.
(will be inserted by the editor)

Identifying Friction between Large-scale Agile Software Development and Plan-based Business Processes

Tobias Altehed · Robin Ingmarsson

Received: date / Accepted: date

Abstract Agile software development has transitioned from small projects to large-scale enterprise settings, incorporating additional influences from lean manufacturing principles such as an end-to-end perspective. At an Ericsson AB development unit, one such process consisting of both agility and planning is deployed in a large-scale setting, having more than 300 developers within more than 30 teams continuously developing software comprising several million lines of code. In this study, thirteen employees of that particular unit participated in interviews, in a workshop and responded to a questionnaire. The collected data was analyzed using an analysis process influenced by grounded theory, resulting in a characterization of friction, i.e. gaps in expected behavior and actual observations between employees in units of the organization working either plan-based or according to agile practices. The results showed that such friction was present between plan-based product management and agile development, as well as between agile development and the plan-based release unit. Furthermore, it was indicated that separation of agile and plan-based organizational units leads to a lack of understanding others' work and valuations, limiting possibilities to optimize across the whole organization. In general, findings indicated a need for improving this end-to-end perspective and it was therefore suggested to incorporate plan-based units into the agile way of working in order to lower barriers to achieve cross-organizational improvements.

Tobias Altehed
Tel.: +46-735-353422
E-mail: altehed@student.chalmers.se

Robin Ingmarsson
Tel.: +46-708-651177
E-mail: ingmarss@student.chalmers.se

Keywords Large-scale · Agile · Plan-based · Friction · Case study

1 Introduction

The agile development methodology for software engineering emerged as a response to manage frequent changes in project scope during the project life cycle [1, 3, 8]. Through dynamic prioritization and focus on feedback, change and teamwork, the incorporation of agile practices has shifted project focus from following a plan to meeting stakeholders' needs at time of delivery.

In parallel to the introduction of agile development, Lean Software Development (LSD) has emerged [22], transferring concepts from lean manufacturing [14, 28] such as an end-to-end perspective, value-stream mapping, inventory management and pull systems to software development [19]. Even though LSD has influenced the business process perspective of agile organizations, project contracts and release plans are for instance still fixed at project initiation due to challenges such as cost and time estimation [7].

While existing empirical studies suggest that an agile process should be adopted by incorporating all relevant practices into the organization [3, 8], finding a balance between agile and plan-based development could improve the alignment between agility in development and planning in business processes, similar to the intension of LSD [4, 5].

For instance, it has been shown that plan-driven management can be successfully combined with agile development [13] and previous research on large-scale agile software development report that organizations using balanced processes have seen improvements in communication, speed and quality [20, 21, 27].

However, it is also reported that adaptations in both agile and plan-based practices are required [7, 12, 13] and in order to gain more insight into large-scale agile settings with the need for adaptation between agile and plan-based processes, additional empirical studies are needed to fully understand how projects are affected by agile practices [10].

This paper specifically sets out to investigate large-scale agile software development by identifying and characterizing friction (See Section 2.2 for definition of friction) between agile and plan-based organizational units in order to assess the effect of friction on end-to-end speed in a software development organization.

The distinction between agility and planning or agile and plan-based is in this paper made as follows; processes containing agile practices or influences from LSD have been grouped into agility, whereas processes based on plans, heavy documentation or stage-gates such as the waterfall model have been grouped into planning. Thus, the two methodologies are not seen as complete opposites and it is not excluded that agility to some extent incorporates planning or vice versa.

As basis for this paper, a case study was conducted by the two authors during January to June 2011. With both authors actively participating at an Ericsson development unit, initial data gathering was performed through document analysis and discussion with assigned mentors, followed by interviews, a workshop and a survey with Ericsson employees.

With the studied unit having both a large-scale context and an existing balance between agility and planning, this study and paper contributes to previous research by providing empirical data on large-scale agile software development in practice. Furthermore, contribution is made by identifying observed friction in balancing agility with planning seen from employees within multiple software engineering roles.

The paper is structured as follows; In Section 2, Streamline Development is introduced together with the definition of friction, followed by a brief review of related work in Section 3. The case study design is then detailed in Section 4 and the results are presented in Section 5 together with the identified friction. Finally, the results are discussed in Section 6 and the paper is concluded by Section 7.

2 Background

In the following subsections, the evolution and introduction of Streamline Development and its relation to agile software development is introduced on a high-level, followed by the definition of friction used in this paper and its intended utilization as an analytical framework.

2.1 Streamline Development at Ericsson

Streamline Development (SD) has been implemented at an Ericsson development unit through several evolutionary steps with a vision of delivering the latest version of the system to any customer at any time.

The extent of agility within SD at another Ericsson unit was by Petersen and Wohlin concluded to be high, with similarities to iterative development, eXtreme programming and Scrum [21]. However, it was noted that not all agile practices was fulfilled, indicating that the SD process is not entirely agile.

More specifically, at the unit selected for this case study, product management and the organizational unit responsible for releasing the product to customers are working traditionally by following a plan. In contrast, the development unit has an agile way of working which incorporates several agile principles such as frequent delivery of working software, co-location and self-organizing teams. This plan-based and agile balance of SD is further detailed in Section 4.1.3.

The introduction of SD at the studied Ericsson unit spanned several years during a step-by-step evolution, which started during the 90's when the development unit employed a waterfall approach for software development. The waterfall way of working was then extended to incorporate anatomy planning, i.e. an approach to avoid late system testing and re-design through feature breakdown, incremental development and test with continuous integration.

The incorporation of anatomy planning eventually forced increased collaboration between disciplines, which led to an introduction of cross-functional teams. Furthermore, the feasibility decision of development projects was divided into several parts, creating early and late phases of the scope to be delivered, thus providing a margin for the development unit to handle small changes late in the project.

When the organization was accommodated with anatomy planning and cross-functional teams, SD was introduced and it is currently employed in 30 teams. So far, three years of incorporating SD during six product releases have resulted in shortened release cycles from eighteen to six months, measurable quality increase toward customers and efficiency improvements.

However, during initial discussions with the Ericsson unit, it was made clear that previous internal studies identified possible bottlenecks, misunderstandings or misalignment between the agile development unit and more traditionally plan-based units of the organization, retaining further improvements from an end-to-end perspective. Given this clear indication of present friction between agility and planning, the Ericsson unit was selected as a case study candidate, strengthened by its large-scale context of many teams developing software comprising several million lines of code delivered to hundreds of customers.

2.2 Definition of Friction

In this paper, the term friction is used as an analytical metaphor to describe and consolidate misconceptions, bottlenecks and problems related to interactions between individuals or roles in a large-scale software engineering organization. This way of using friction as a metaphor from physics to other scientific subjects is nothing new and has previously been summarized by Åkerman [2].

More specifically, in the framework used for analysis in this paper, friction is seen as the gap between roles' or individuals' expectations and actual observations of each others' actions as illustrated in Figure 1.

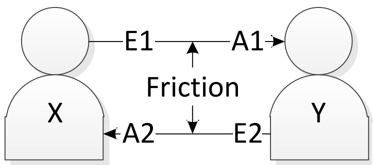

Fig. 1 Friction is defined as the gap between roles' or individuals' expectations and actual observations of each others' actions

For example, individual X in Figure 1 observes friction in the personal or professional relationship toward individual Y if the expectation (E1) on how individual Y should act does not correlate with how X observes that Y actually acts (A1). Simultaneously, individual Y can expect (E2) and observe actual behavior (A2) of individual X, resulting in additional friction if a mismatch exists.

With this simple analytical framework, statements captured during interviews regarding the current and desired way of working can be structured to effectively illustrate where in the organization misconceptions, bottlenecks or problems occur. The illustration can then be used to provide a concrete base for further improvement work, as explicit statements on expectations and observations can be used to facilitate constructive discussions across the organization.

3 Related Work

In pace with the rise of agile practices [3, 8, 11] for software development, several empirical studies have been conducted in industry to determine the effects of increased agility. For instance, Dybå and Dingsøyr provide a literature review of 36 high quality empirical studies [10], in which they identify both improvements and limitations of agile development.

In addition to an overall agile mindset, SD has clearly taken influences from LSD such as the end-to-end perspective or as specified by Poppendieck and Poppendieck; optimizing the whole [22]. Mehta et al. [16] provide insight into how lean principles relate to a real software development context, but in contrast to case studies only covering agile practices the extent of LSD empirical studies is limited. Apart from Middleton who provides a good coverage of LSD evolution [17] and the first empirical results of a LSD implementation [18].

With Ericsson being a large company where the studied unit has rolled-out SD to 300 developers in 30 teams, a large-scale perspective on agile development must be taken into consideration. Therefore the next section will detail case studies of large-scale agile implementations and other studies of interest relating to this thesis.

3.1 Large-scale Agile Case Studies

Petersen and Wohlin conducted an interview study at another Ericsson development unit, where perceived improvements after transitioning from a plan-based way of working to SD were investigated [21]. It was concluded that the agile transition eliminated common issues from plan-based development and that agile practices led to increased release frequency, reduced lead-time, improved quality and communication while enabling a reduction of waste and change requests. However, they state that current challenges exist, including an increased need for test coordination and test coverage, a need for support of team-coordination and decision making for concurrent projects and a need to integrate release projects in the development process.

Furthermore, Petersen and Wohlin recognized that the majority of past research concerning agile software development focused on one specific model and small projects [20]. They therefore re-analyzed data from the same interview study, focusing on identifying advantages and disadvantages comparing the findings to previous empirical research in order to evaluate whether or not large-scale projects receive the same advantages and suffers from the same disadvantages as small-scale projects. It was concluded that most of the advantages gained by using agile practices in small-scale projects are directly transferable to large-scale projects. For instance, communication in teams reduces the need for documentation and early feedback is possible thanks to frequent delivery and reduced amount of rework. In contrast, few of the identified issues with large-scale development had been found in previous research regarding small-scale projects. For instance, large-scale agility brought challenges regarding continuous testing, increased configuration management effort, increase in product-packaging and release effort and increased maintenance effort due to the increasing number of releases.

Another study made by Tomaszewski et al. presents an evaluation of the suitability and applicability of SD for Ericsson [27]. The evaluation was performed by identifying advantages, disadvantages and potential issues related to an SD process implementation using data collection from 27 interviews with Ericsson employees. The findings were classified into three categories; things that would improve, worsen or need to be changed in an SD introduction. It was concluded that SD has great potential, recognizing that there were many things that would improve with the process, e.g. increased motivation of the staff, increased customer responsiveness, simplified maintenance (in contrast to the findings presented by Petersen and Wohlin [20]) and improved communication. On the other hand, some that would worsen were also identified, e.g. difficulty in assuring quality and long-term architecture deterioration. However, it was concluded that many of these could be avoided if necessary changes were made prior to introduction, e.g. introducing continuous requirements management, increase testing efficiency, improve architecture and increase the efficiency of the installation procedure.

In relation to partially agile and partially plan-based processes such as SD, Karlström and Runeson investigate how an agile development process fit together with stage-gate management. By studying large-scale development settings at ABB, Ericsson and Vodafone where agile development had influenced or been integrated into plan-based management models [12, 13] they found that both methodologies gave tools to each other, i.e. agility improved planning by having a day-to-day focus and planning improved agility by providing means for inter-team coordination and communication. However, the authors stress that adaptation of the methodologies to be combined is required and that a cross-organization acceptance of the employed process is a major success factor.

A similar conclusion is supported by Lindvall et al. who analyzed experiences from pilot projects of agile introduction at four large organizations; ABB, DaimlerChrysler, Motorola and Nokia [15]. They found the greatest challenge to be integration of the agile practices into the existing process, requiring tailoring to integrate agile projects into the organization rather than agile practices into a project. Concluding that clarifying and specifying the interface between agile and non-agile parts of the organization is required to minimize the need for re-work and misunderstandings.

The need for adaptation of agile practices to better fit large-scale software development organizations is also recognized by Cao et al. [6]. Through an industrial case study they identify key differences when agile is employed in a large setting, such as up-front creation of a stable architecture and to surrogate customer involvement to reduce problems with an extensive domain being too much for a single customer to have detailed knowledge of.

Talby et al. conducted a case study at the Israeli Air Force focusing on how to work in an agile manner with quality and testing in a large-scale software project [26].

It was emphasized that defects should be resolved as soon as they are discovered in order to enable delivery of working and stable software to the customer at every iteration. They concluded that this way of working with defects has a set of advantages. First, the effort to fix a defect is significantly reduced. Second, development pace can be increased with a clean and stable baseline. Third, it eliminates the overhead of defect prioritization and planning, making negotiation with the customer over which defects to fix unnecessary.

Sutherland describes his experience from working in five different projects and organizations where agile software development, and Scrum in particular, has been introduced and successfully used [25]. In addition to the conclusion that Scrum works in any environment and is scalable to large projects, it is recognized that development speed can be significantly increased as a direct effect of communication and information sharing. Reflections from one project state that people were aware of what other team members were doing and could therefore work in a way that were beneficial for the whole team, ultimately eliminating much work for the team members.

On the topic of balancing agile software development with plan-based business and release, Boehm and Turner created a framework for adapted projects [5]. They recognized that both agile and plan-based methods have short-comings which if unaddressed can lead to project failure. The framework constitutes a five step model where risk is used for structuring projects in order to incorporate both agile and plan-based practices. First, risk analysis is used to specify risk areas in relation to agile and plan-based methods, categorized into environmental, agile and plan-driven risks.

Second, the risks are evaluated to determine whether or not the project should use a pure agile or pure plan-based methodology. Third, given that the project is unsuitable for a purely agile or plan-based approach, an architecture is developed that supports agile methods where their strengths are exploited and weaknesses minimized. Fourth, a risk management and resolution strategy is developed. Finally, project attributes are monitored in order to enable proper adjustments toward either agility or planning during the project life time.

From the related work, several findings contribute to this study. First, the studies conducted at other Ericsson development units provide a direct comparison basis with details on the agility of SD, effects of the transition and improvement recommendations. Second, studies of other large-scale or balanced software development processes contribute with additional findings such as alternative solutions along with their positive and negative effects. Third, specific studies focusing on testing, communication and risk assessment provide additional insight and contrasting material to behavioral aspects or technical details in the organization. Furthermore, the design of the related studies has provided input to the design of this case study with respect to interviewee selection, interview questions and validation techniques.

4 Case Study Design

In the following subsections, the context of the studied project, product and process at Ericsson is detailed. Furthermore, specifics regarding the study design are explained, including research questions investigated and data collection and analysis procedures used.

4.1 Study Context

This case study has been conducted at Ericsson AB, a multinational telecommunication and data communication company developing various mobile and fixed network products. With a market share of 35 percent and over 90,000 employees in more than 150 countries, Ericsson is the world's largest mobile telecommunications equipment vendor and the world's fifth largest software development company.

4.1.1 Project

The studied project consists of approximately 300 developers distributed over 30 teams, serving hundreds of customers with a target of delivering high quality with dependability. The teams are cross-functional and co-located to enable close communication between developers and teams but also with project management and release units.

Overall, the project organization can be divided into three parts; Product Management (PM), Research and Development (R&D) and Product Introduction, Deployment and Support (PIDS). Within both R&D and PIDS, roles include program leader, system manager, team leader, designer and tester and the employees taking these roles belong to a line organization in parallel to the project but usually devotes all their time to the project. Simplified, R&D receives input from PM to develop, test and integrate new features that are transferred to PIDS in order to be released to customers and thereafter maintained and supported.

How information flows between the different parts of the organization and its customers is illustrated in Figure 2. The relation between PM and customers includes management of requirements (1) and strategical issues (2), e.g. negotiations regarding what other suppliers and Ericsson can deliver. Usually, R&D and customers do not communicate directly, instead the information flows either through PM or PIDS depending on the matter; if a customer wants to know more about specific features and how they work (3) or if R&D wants more information from a customer regarding requested features or trouble reports to be corrected (4). Additionally, PIDS selects a collaboration customer (5), executes live tests on customer site and finally puts the release out for general availability to all customers (6).

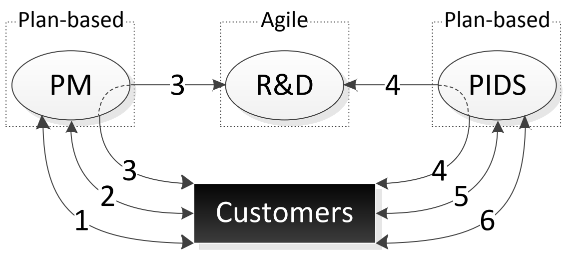

Fig. 2 An overview of the organization and its relation to customers

4.1.2 Product

The project develops two separate mobile network units consisting of both hardware and software, acting as gateways between mobile devices and the Internet. However, the two units provides functionality in different generations of mobile telecommunication, complementing each other and are therefore usually combined into a single entity.

The product has been in development since the mid 90's, first released commercially in 2001. The software is developed separately from the hardware development and several releases of software are made before a hardware upgrade of the unit occurs. The software has been updated continuously and is currently subject to two releases per year following the SD process described in detail in Section 4.1.3.

In total, the software comprises several million lines of code, divided equally between C and Erlang [9] source code. Several different releases are currently maintained and supported through a number of correction packages sent to customers containing defect corrections.

4.1.3 Process

In contrast to previous employed processes at Ericsson, SD has ensured that important decisions are no longer made early and upfront; rather they are made just in time through a conscious focus on speed to shorten lead times. High speed is seen as delivering small increments of added customer value (cf. single piece flow [28]) by integrating working parts of a feature, i.e. anatoms, to a branch of the software compromising the latest system version (LSV).

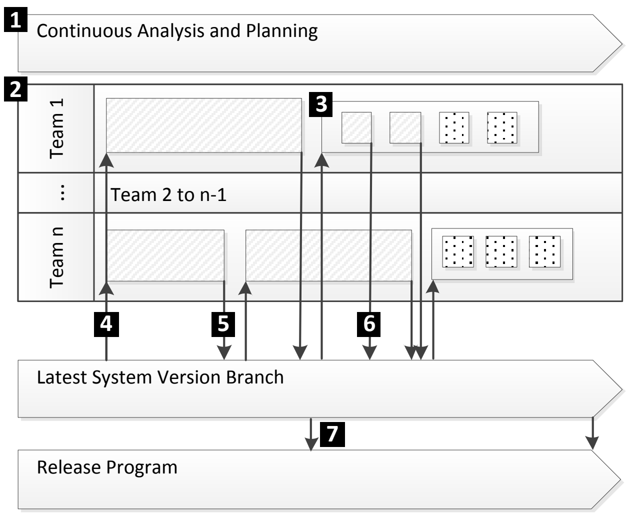

Fig. 3 Overview of the Streamline Development process

From an overall and purely theoretical perspective, the SD process (See Figure 3) includes continuous analysis and planning with feature prioritization (1), submission of features to cross-functional teams just-in-time (2), division of features into anatoms (3), continuous renewal of developers' code-base (4) and integration of each completed feature (5) and each completed anatom to the LSV (6) while having the release program separated from development (7).

More specifically, the SD process is in theory guided by five cornerstones;

I. The highest priority feature is assigned to a team when available

The decision to develop a new feature is taken by a continuous analysis and planning program as soon as a development team is available. The program planning maintains a prioritized list of features, which is analyzed and re-prioritized bi-weekly. Each feature is specified to the extent where there is just enough information for it to be handed over to a development team.

II. A team develops the feature during an average of three months with full responsibility of its implementation

The cross-functional teams, having an average of seven employees, incorporate roles such as system managers, designers, developers and testers. Each team is designated to develop and deliver one feature at a time during roughly three month intervals, with full responsibility from end-to-end of the development, i.e. from pre-study through system test and also if needed through network integration and verification.

III. The team breaks down the feature into smaller parts, specified in an anatomy plan

Upon receiving a feature from program planning, the team is supposed to break it down into smaller parts called anatoms to be delivered sequentially during the three months of development. The team can either break the feature down into anatoms upfront or decide to specify only the first anatom and defer the specification of the latter parts of the feature to be specified when more knowledge is acquired during the implementation of the first anatom.

IV. Each completed anatom is integrated to the LSV

Regardless of how the feature is divided into anatoms, the team is expected to deliver and integrate a working part, i.e. one anatom, of the feature to the LSV each month. The LSV should therefore represent the latest state of the software, making it possible for all teams to use the same code base. Eventually, with 30 teams integrating small functional pieces every month, the lead time for the release projects should be shortened while the software grows in small steps.

V. Release of the LSV to customers is decoupled from the development organization

PIDS follow their own stage-gate project process deliberately separated from the agile R&D organization due to legacy reasons, i.e. to maintain a plan-based way of working toward customers while enabling R&D to transition to an agile way of working in order to increase development speed.

Figure 4 illustrates the plan-based release process employed in PIDS, starting with product packaging (PP) to set prices on features or packages of features in the new release, securing supply of hardware supporting the release and finding one specific customer to involve in first office application (FOA). When a pre-release assessment (PRA) has been conducted to ensure that the software is ready to be released, the content in the form of code, supporting models and documents at the LSV is delivered from R&D to PIDS.

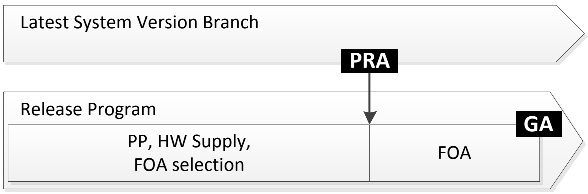

Fig. 4 An overview of the plan-based release process

Following that delivery, the actual FOA work is engaged and planned to last for eleven weeks, including a three week clean run where the test object list is executed without customer participation, a four week acceptance test with the customer and a four week primary consolidation integrating the node with the new release in the customer's network. After a successful FOA, the release is set to general availability (GA) for all customers.

When GA has been set, the specific release is branched into a separate maintenance track handled by PIDS, where trouble reports are solved and defects are corrected.

4.2 Research Objective and Questions

This work strives to provide desired empirical insight on large-scale agile development in general and balancing agility with planning in particular.

In the following, a set of research questions (RQ) and sub-questions (SQ) to be investigated in accordance to the objective is listed. RQ1 and its SQs are strongly related to the definition of friction and its categorization while RQ2 and its SQs are more related to assessing the effect of friction on the organizational performance.

Even though SD has resulted in improvements to the work performed at the studied Ericsson unit, the decision to focus on identifying friction and its effects was made in order to provide the organization with concrete instances of possible improvement areas.

- **RQ1:** *What characterizes the friction between agile development and plan-based business?*
  - **SQ1.1:** *What is the current expectation on the way of working between individuals and roles in different units of the organization?*
  - **SQ1.2:** *What is the actual or observed way of working in different organizational units?*
- **RQ2:** *What is the effect of friction on the end-to-end speed in the organization?*
  - **SQ2.1:** *How has the friction affected the organizational units?*
  - **SQ2.2:** *To what extent are the organizational units aware of the effects?*

4.3 Data Collection

The data collection followed a general case study approach, i.e. an in-depth investigation of a single study object, chosen due to its suitability for software engineering research in a natural context and the incorporation of multiple elements of research methods, being both exploratory and qualitative as described below [23].

During the study, interviews were conducted with employees, supported by observations at internal workshops, guest lectures held by Ericsson employees and continuous study of archival data such as process documentation. All data sources were strategically selected in order to ensure data triangulation, i.e. making it possible to cross-reference findings. In the following subsections, these data sources are described in further detail.

4.3.1 First Degree Data Sources

As a first degree data source, interviews were conducted where interviewees were strategically selected to collectively provide a broad perspective covering all phases of the project, i.e. spanning from PM through R&D and PIDS. The selection was also made to provide customer insight from interviewees working in roles having direct contact with Ericsson's customers. Furthermore, most interviewees had several years of experience from working at Ericsson, being able to follow the evolution of SD, often through working in different roles historically. The distribution of their roles is summarized in Table 1.

Table 1 Distribution of interviewees

Org.	Role
PM	Product Manager
PM	Product Release Responsible
R&D	Manager
R&D	System Manager and Team Leader
R&D	Developer and Designer
R&D	Developer and Function Tester
R&D	Function Tester
PIDS	Release Program Manage
PIDS	Release Execution Team Responsible
PIDS	Release Responsible
PIDS	Product Packaging and Supply
PIDS	Customer Product Information
PIDS	System Tester

As seen from Table 1, the distribution of interviewees was uneven in relation to the number of organizational units. However, interviewees were deliberately selected to cover all roles with relevant responsibilities for the study within each organizational unit rather than triangulating each role. Thus, the distribution of interviewees is strongly related to the roles within the studied project, i.e. more interviewees were selected from PIDS than PM since there were more relevant roles to be interviewed in that organizational unit for the purpose of this study.

The interviews were between 30 and 75 minutes long, following a semi-structured interview guide as outlined in Table 2 with the purpose of in-depth investigation of friction between agility and planning. When an actual instance of friction was discussed, follow-up questions were posed to the interviewee in order to reach the causal factors in relation to possible gaps between expectations and actual behavior. At the end of each interview, the interviewee was asked to rank the three most problematic friction instances and provide own improvement suggestions.

For example, during the Q3 section of Table 2 the question *"Has the vision of delivering to any customer at any time affected your own work and interaction with other departments?"* was asked, followed by *"Is it consistent with your expectation on how work and interaction between different parts of the organization should be performed?"* and during Q5 the question *"How do you think that the vision of delivering to any customer at any time involves the customer?"* was followed by *"What is your perception of other employees' opinions regarding this?"*. Thus, the interviewees were first declaring their observed friction within each section and then further asked to detail their observations in a way that corresponds to the definition of friction and its use as a framework for analysis.

Table 2 Structure of interview guide

Section	Topic
A	Information on anonymity and recordings
B	Description of study purpose
C	Set the focus on agility and planning balance
D	Explain commonly used concepts
Q1	Formal role and performed activities
Q2	Observed friction in general
Q3	Friction in relation to the way of working
Q4	Friction in relation to organizational structure
Q5	Friction in relation to external forces
Q6	Prioritizing friction and improvements
Q7	Open-ended questions

All interviews were conducted by both authors, recorded, transcribed word by word and sent to the interviewee for review and validation in summarized form. In total, 13 employees were interviewed; 2 from PM, 5 from R&D and 6 from PIDS, resulting in 10 hours of recordings and over 140 pages of transcriptions.

4.3.2 Second and Third Degree Data Sources

In addition to interviews, data was collected through discussions with employees from PM, R&D and PIDS. Specifically, employees having an overview of processes and work performed within the organizational units presented high-level descriptions for basic understanding of the current way of working.

Furthermore, an internal workshop relating to bottlenecks in the SD end-to-end flow was attended. Here, employees who previously participated in an interview study gathered to discuss bottlenecks and possible enablers to remove them. From this workshop, an initial understanding to possible friction areas was given, ultimately forming a basis for construction of the semi-structured interview guide.

As a complement to spoken information, archival data was used for understanding the organization and its process. For instance, process documentations and checklists were investigated to deepen the understanding of the way of working and terminology used within the studied unit and project.

4.4 Data Analysis

While second and third degree data sources were mostly used to provide a general understanding of the current state in the organization and to indicate possible friction, the interviews were thoroughly analyzed in order to extract and compare explicit statements related to friction and the underlying causes identified by the interviewees.

The process used for this analysis followed basic principles from grounded theory [24], modified and extended with a validation workshop and survey as outlined in Figure 5.

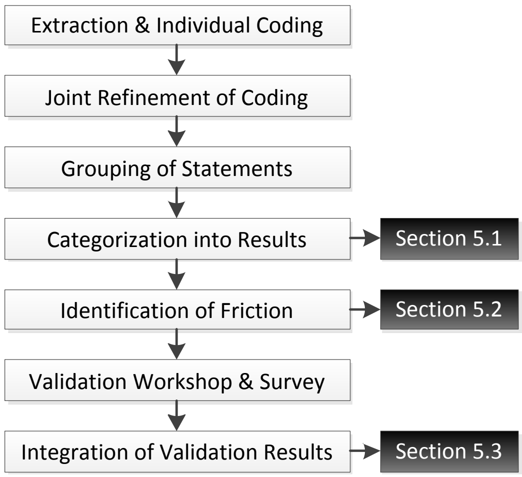

Fig. 5 Data analysis process and related sections

Figure 5 shows how the word-by-word transcripts were analyzed by the interviewers, starting with three steps. First, relevant statements from three of the transcribed interviews were extracted and assigned a code by each interviewer individually [24]. Second, the individual coding was refined through joint cooperation where deviation in coding and non-overlapping in extracted statements were taken into consideration [24]. Finally, the extracted statements were collected into groups containing statements on the same subject [24].

This process was then repeated for the ten remaining transcribed interviews, although with an established base of categories the extraction and coding of statements was done individually, dividing the ten interviews equally between the interviewers.

With an established set of groups, further refinement into categories was performed jointly, resulting in the set of four results presented in Section 5.1. From these results, specific instances of friction between roles or individuals were analyzed with respect to interview findings and related work, the results of which are presented in Section 5.2.

Finally, the friction instances were discussed at a validation workshop complemented by a validation questionnaire, which were conducted together with the 13 interviewees. The workshop and questionnaire focused on validating that each friction exists, whether or not it is related to a balance of agility and planning and if the friction was present before SD was introduced. The result of the survey is presented in Section 5.3.

4.5 Validity Threats

According to grounded theory, threats are divided into four categorizes; fit, relevance, workability and modifiability [24]. However, in software engineering studies, threats are conventionally divided into the four categories that will be used here; construct validity, internal validity, external validity and reliability [23]. First, construct validity concerns the method of collecting the most valuable and accurate data for the study to be conducted. Second, internal validity primarily concerns explanatory and casual studies. However, this study is of exploratory and qualitative nature, thus internal validity is not considered. Third, external validity concerns the degree to which the study findings can be generalized outside the study context. Finally, reliability concerns whether or not the study can be repeated or replicated producing the same result.

4.5.1 Construct Validity

The interview results are highly dependent on the selection of interviewees. In order to obtain the best possible sample, the selection of interviewees was made in close collaboration with experienced Ericsson employees, having extensive knowledge about the organization and the employees as well as experience from conducting similar internal studies.

Since both interviewers were external researchers, there is a risk that the interviewees' answers were influenced. However, with support from experienced Ericsson researchers and given that it was a continuation on previous research, it was not perceived as a large risk. In addition, the interviews were guaranteed anonymity in order to further mitigate the risk.

The risk of the interviewees misunderstanding the questions or the interviewers misinterpreting the answers has been mitigated through several actions. First, the purpose of the study and all concepts used during interviews were explained to all interviewees prior the interviews. Second, interview questions were compiled in close collaboration with researchers and employees internally at Ericsson as well as with external researchers. Finally, all interviews were recorded, enabling the interviewers to follow up interviews for clarification and thereby resolve possible misinterpretations.

4.5.2 External Validity

Due to the limited time span of the case study, it has been conducted at a single company, posing a potential threat to the generalizability. In order to mitigate the risk from the study being influenced by this, general findings from previous research within the field of software engineering was mapped to the findings from the studied Ericsson unit, thereby enabling general conclusions to be drawn.

4.5.3 Reliability

In order to mitigate the risk of the interview answers being misinterpreted by the researchers, data has been collected from multiple sources, i.e. triangulated to ensure the correctness of the findings, and validated both individually and collectively. First, documents describing the current processes were studied, summarized and validated in order to have a complete understanding of the theoretical process prior to the interviews. Second, all recorded interviews were transcribed, summarized and sent to the respective interviewee for individual validation and comments on the interviewers' interpretations. Third, a workshop was held together with all interviewees, discussing and validating the findings. However, due to time constraints, only four frictions were discussed during the workshop; F2, F3, F5 and F6. To compensate for this, a survey was conducted prior to the workshop using a questionnaire in order to cover those frictions omitted from the workshop, while also further triangulating the frictions included in the workshop.

5 Results

Through the data analysis process used, interview findings were ultimately consolidated to four major result categories. These findings are presented in the next subsection, followed by the identified friction instances derived from the categories and their relevance for balancing agility and planning. The section is then concluded with the findings and analysis of the conducted validation workshop and survey.

It should be noticed that this study is aiming at identifying potential friction from using a balanced process of agility and planning. With this focus on friction, the results tend to seem negative. However, the SD transition has generally provided substantial improvements throughout the organization at the studied Ericsson unit. Despite that, this section only reflects momentary issues, it does not go into depth regarding general improvements or possible issues related to the transition, it only focuses on the current balance between agility and planning. Furthermore, the results presented in the following sections are highly dependent on a small sample of interviewees in relation to the size of the organization and their subjective opinions on the way of working. The presented result should therefore not be considered as fact but rather individual views and opinions grouped together to describe the current situation from multiple perspectives.

5.1 Interview Findings

Findings from the interviews have been consolidated into four result categories, R1-R4, as summarized in Table 3 and further detailed below.

R1. Undefined communication interfaces

Interviewees agreed that it is often unclear who to contact regarding a specific errand, i.e. communication interfaces in the organization are partly undefined. Mostly, interviewees stated that they rely on previous contacts and experiences when a need for communication arises, leading to a self-organized and informal exchange of services and favors. In general, self-organizing within agile software development is desired [8], but in this specific case unwanted effects have been noted.

For example, the relation between PM and developers in R&D is occasionally informal, even though SD specifies formal ways of communicating, i.e. cornerstone 1-3. This informality has resulted in unclarity regarding on what grounds PM base promises toward customers on and what grounds developers base promises to PM on. It has also resulted in unclarity regarding responsibilities associated with the respective roles.

Specifically, developers tend to contact PM regarding development specific issues, which PM consider not to be their responsibility to answer since they do not want to become an internal source of requirements with the risk of loosing customer focus, leading to a development organization without any market perception. Instead, it is desired by PM that the R&D organization should be elevated to a level where they can solve their specific issues without PM involvement.

Another example of this informality is that resources performing work in PIDS are usually taken from teams in R&D since the PIDS organization is too small and lack sufficient competence to correct errors and fix bugs. For this purpose, a formal interface for resource allocation exists but still the informal contacts are faster and therefore more often used, resulting in uncertainty regarding what resources are available at a specific point in time. In the end, this affects the flow end-to-end in the organization; PM cannot be asserted on a committed scope since R&D cannot know whether developers are working on new feature development or fixing bugs, risking delayed delivery affecting PIDS's time plan.

This unclear interface between R&D and PIDS is otherwise apparent regarding the handover at PRA.

Table 3 Summary of interview findings

ID	Description
R1	**Undefined communication interfaces** between agile and plan-based parts of the organization have led to unclear areas of responsibility
R2	The deliberate isolation of agile organizational units limits **possibilities for understanding** how one's work affects other parts of the organization
R3	Agile and plan-based parts of the organization have **different time and quality limitations**
R4	The **desired extent of customer involvement** differs between agile and plan-based parts of the organization

Here, the expectations on the details of what to be delivered differ between R&D and PIDS. For instance, R&D realize that PIDS have a need for documentation, but it is hard for them to know in detail what to deliver. At the same time, PIDS realize that R&D need and want quick feedback, but find it hard to locate a specific person or role to use as a feedback channel.

As a possible effect, the responsibility to update documents describing the processes had been lost in the gap between R&D and PIDS, which led to documents containing erroneous data concerning agreements of hand-over dates from the agile R&D organization to the plan-based PIDS organization. PIDS were still relying on the agreements stated in the documents while R&D in their new agile way of working had no possibility to commit to these fixed hand-over dates as before the transition.

Furthermore, the unclear interface has led to responsibilities belonging to R&D being taken care of by employees in the PIDS organization. PIDS argue that quality assurance work and responsibility that should rest on developers or teams in R&D now have been informally put on PIDS. However, R&D is utilizing the possibility created by SD to incorporate features late in parallel to PIDS's FOA work in order to achieve faster delivery rather than delaying the whole scope.

R2. Possibilities for understanding

PM, R&D and PIDS have deliberately been built up as separate units of the organization, most noteably through PIDS's separation in accordance with SD's fifth cornerstone. The conducted interviews show that this has led to a separation of decision making with an unwanted effect on understanding the importance and impact of work performed by others.

Apparently, existing dependencies between the units have become hard to see from the employee perspective in all different parts of the organization. In general, employees are not aware of how their own work affect others', a finding that is well rooted in R&D who argue that PIDS do not know how SD works in detail as well as in PIDS who argue that R&D do not know what happens after development or how the release and FOA procedures are executed.

Within R&D, the general opinion is that it is clearly stated that development teams should be isolated, which from their own perspective is perceived as beneficial since they can focus on developing features with high quality without being disturbed. At the same time, they want to receive adequate information from the surrounding world both internally from Ericsson and externally from the market.

From the PM perspective, it is clear that the development teams in R&D want stability and continuity while receiving requirements to develop features without interference from PM. However, it is desired from teams in R&D that PM is present at the start-up meeting for new feature development in accordance with the SD process in order to give another perspective on the often very technical requirements mediated by system managers.

From the R&D perspective, the separation between R&D and PIDS is seen as a proof that R&D does not need to have any contact with PIDS since they do their own things. However, from PIDS perspective this is seen as if incentives for delivering features on time have been removed, since without contact with people affected by delays, developers will not understand why delivering on time is crucial for other parts of the organization.

Employees in the PIDS organization blame R&D for delaying the agreed date for PRA, which in fact has been delayed several months on occasions. PIDS does not have insight in the difficulties with breaking down features to deliverable anatoms and is therefore experiencing that R&D has been given a possibility to not deliver on time according to the principles of SD, stating that a committed feature can be delayed and re-committed to a later release or canceled altogether. Thus, plan-based organizational units see the desire of totally isolated teams as a dangerous mindset, arguing that it is important for developers to not believe that SD has been introduced to provide tools enabling them to delay features rather than increasing speed through development.

Specifically, it is assumed by developers that their respective team leader should communicate delays and warn for possible late deliveries toward higher managers in the R&D line organization in accordance with the formal process. However, escalation from the individual developer through team leaders and managers toward PIDS does not always occur early enough, which PIDS blame the team isolation for since it defers responsibility for the delayed delivery outside the team sphere, making the team unable to understand the consequences that arise for PIDS when information regarding delays does not reach them in time.

In the end, the lack of understanding how others' work is affected by one's actions has led to several misunderstandings reoccurring in the daily work. First, the organization is never learning from delays occurring at PRA because of the achievement of delivering high quality at GA, making occasional delays at PRA appear less important and reducing the experienced need for feedback. However, by not improving or receiving feedback on the effects of delaying PRA, the scheduled time for validation during FOA shrinks, putting higher load on the PIDS organization to do their work on shorter time. Second, PM and developers make promises without consulting PIDS on how the promises can be realized during the release process, forgetting that testing, integration and verification should be included in promises to the customers. Third, PIDS are requesting information and documents from R&D which they can no longer deliver early due to their agile way of working where documentation is done after a feature has been completed.

R3. Different time and quality limitations

At the incoming end of the studied unit, PM exhibits pressure regarding delivery date and scope from the customer, which is passed on to R&D. Simultaneously, at the outgoing end of the studied unit, PIDS exhibits pressure from customers regarding both time and quality of deliveries, which in turn is passed on to R&D. Thus, different parts of the organization experience pressure from different directions, resulting in priorities differing between organizational units.

For instance, it was expressed explicitly by interviewees that meeting a delivery deadline has higher priority than continuing development of features for the next release. Additionally, R&D expressed that they understand that PM has to promise to deliver a set scope at a certain point in time due to the selling pressure they are under, otherwise they will be outrivaled by Ericsson's competitors.

R&D's perception is that most often is time of highest priority and feature content is of less importance as long as deadline is met and delivery is made on time. Interviewees stated that a feature delayed at PRA can be included in the release during FOA since several workarounds can be utilized. The feature can either be patched in, be added to a new build just prior to general availability or decreased in scope. While seen by PM and R&D as a possibility to incorporate features late for quick delivery to customers, the experience within PIDS is that this affects their time-plan by creating additional work.

Similar backups do not exist for PIDS, who are tightly bound to meet the assigned time slot at the customer site during FOA. It is therefore argued by PIDS that the date for general availability is prioritized over the date for PRA since the deadline toward customers must be met in order to maintain a leading market position, but the deadline toward PIDS can be stretched. The result of this is that PIDS experience reduced quality during FOA, having more trouble reports to be corrected and extra work to be conducted during a shorter time span than planned.

In addition, it has been stated that the last delivery prior to PRA is usually not an increment of the feature, rather the entire feature, which is argued to be a causal factor to delayed PRA. This problem stem from a series of difficulties that the developers have to cope with. First, because of problems with integration and different test scopes on the development branch and the main branch, developers know that quality on the development branch is irrelevant. No matter the level of quality, integration always results in problems which cannot be traced back to its derivation. Second, teams have not become accommodated with feature breakdown, resulting in deliveries of features rather than anatoms in contrast to what is stated in the SD cornerstones.

Consequently, the main branch is not of desired quality, which in turn means that there are many defects to correct at the end of a development cycle in order to reach the quality level expected by customers. This has therefore created a quality gap between preliminary builds and builds going out to customers. Although it has been stated that delayed general availability date is unacceptable, the high concentration of integrations close to the last date for delivery has a direct effect on the possibility to be ready for general availability, mostly stressing PIDS who cannot utilize any workarounds to compensate delays.

R4. Desired extent of customer involvement

The customer requirements are passed from customers to developers in R&D through PM and system managers and can therefore sometimes become distorted on the way. Although a feature has been developed according to specification, PM sometimes misinterprets what the customer actually means, which may result in delivery of a feature not matching the customer's actual needs. However, employees in R&D argue that this could be avoided by having initial customer involvement when starting the development of new features. Customers would then be able to explain their actual needs directly to developers, give feedback on possible solution suggestions and answer any questions on implementation details where PM lack competence, ultimately limiting the risk of developing the wrong feature or developing the feature incorrectly.

Although developers have stated that more customer involvement would be desirable, PM is under the impression that R&D wants stability and continuity without disturbance from customers. Furthermore, PM does not allow customers to directly affect the development environment where teams develop features for the mass market. In the large-scale setting of Ericsson, obtaining too much influence by one close customer relation out of hundreds possible is by PM argued to be undesired since the feature might become too customer specific, serving one rather than all customers.

Apart from sometimes misinterpreted requirements, too low direct involvement has led to the development unit not acquiring adequate information regarding why a customer wants to participate in FOA and what is important for that particular customer. Since a particular test scope is set for each FOA, this ultimately problematizes prioritization of features and trouble reports during development. Because of this, all features do not always get acceptance tested, which produces much additional work afterwards for PIDS. However, R&D states that with better knowledge regarding the FOA customer and the reason for FOA, priorities during development would improve.

5.2 Identified Friction

Apart from assessing the current state of balance between agility and planning, the results in the previous section were further analyzed in order to characterize friction according to the definition in Section 2.2. In total, seven instances of friction were identified between agile and plan-based parts of the organization (See Figure 6).

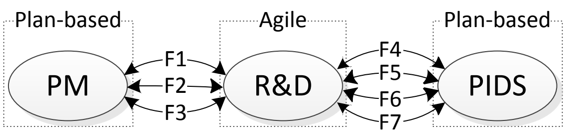

Fig. 6 Friction is present between agile and plan-based parts of the organization

The seven friction instances are elaborated below, complemented with explicit statements from the conducted interviews that highlights expectations and actual observations spoken from employees in PM, R&D and PIDS.

F1. Friction in the responsibilities to detail customer requirements

It has been shown from the interviews that it is unclear where in the organization the responsibility lies for providing implementation details to the development teams. This friction is derived from the result R1 and is further concretized through the following quotes:

> *"There is an expectation within R&D that PM should feed them with requirements specific enough for implementation"* (From PM)

> *"R&D must elevate themselves, fill in blanks and investigate fuzzy requirements on their own"* (From PM)

While R&D expect PM to answer questions regarding development details, PM lack detailed implementation knowledge and do not want to be an internal source of requirements since they fear loosing customer focus and market perspective due to increased workload. PM therefore expect R&D to elevate to a level where they can resolve implementation details without PM involvement. However, with the current situation of isolation of R&D and the lack of sufficient customer involvement, PM experience that R&D actually await input rather than taking action on their own.

F2. Friction in the degree of customer involvement

From the interview results, it was clear that PM and R&D had different perspectives on the degree of customer involvement, i.e. R4, and that teams in R&D expect PM to be present at start-up meeting for features, i.e. from R2, two aspects that are illustrated by the quotes below:

> *"Customers should not be directly involved in development since that will give very few key-customers too much influence"* (From PM)

> *"It was decided that PM should attend start-up meetings for features, but I have not experienced that yet"* (From R&D)

Here, friction has appeared since teams expect as much customer involvement as possible in order to acquire adequate information to develop the correct feature with high speed. However, PM do not want customers to be directly involved in features that are developed for the mass-market since it will give them too much influence, which is illustrated by the first quote.

As a current trade-off, PM is expected to act as a mediator of customer needs in parallel to system leaders mediation of technical requirement descriptions, but R&D experience that PM actually are not present at the start-up meetings as agreed which is illustrated by the second quote. This spurs R&D to request direct customer contact, contrasting the view from PM and creating friction in the understanding between the roles.

F3. Friction in committing to a scope or allowing the time needed by teams

The result in R3 clearly shows how R&D understands that PM has to commit to a scope. However, it also shows that R&D still experience that the development unit is under large pressure due to the promised scope being too large, which is illustrated by the following quote:

> *"PM have sold something that customers expect at a certain time, but sometimes we should allow the time it takes in order to develop a feature"* (From R&D)

While there is a mutual understanding that a scope has to be set despite the agile way of working within R&D, their expectation on a feasible scope is not always experienced to be met by PM, occasionally resulting in a feeling of overload within R&D. At the same time, PM expects R&D to commit to delivering features at a certain time, but with the high pressure in R&D it is hard for teams to commit. Thus friction is created in the relation, occasionally resulting in that PM bypasses the formal structure set by SD in order to get teams to commit to deliveries, indicating that PM has a need to be more agile than the current process allows.

F4. Friction in understanding others' way of working

Following the result in R2, it is clear that there is a lack of understanding regarding others' way of working and how one's own work affect other parts of the organization. Specifically, the deliberate separation between R&D and PIDS has created barriers between them, which is illustrated in the following quotes:

> *"I believe that PIDS does not have any real insight regarding how SD works, at least not in detail"* (From R&D)

> *"Employees in R&D do not have a clear picture of work performed after development; they see it as somebody else taking care of post development matters"* (From PIDS)

The perception within R&D is that the isolation is beneficial, leading to more focused development and thereby higher quality. Howeverm, PIDS experience that R&D want to be isolated and separated from PIDS but still expect R&D to understand how their agile way of working affect PIDS. In contrast, R&D expect PIDS to understand the development context of SD, but experience that PIDS have no detailed understanding of R&D's way of working.

Thus, friction is present between R&D and PIDS, since both organizational units expects mutual understanding of the way of working, but actually experience that there is no interest in understanding how others work.

F5. Friction in incentives for delivering on time

From R2, it was identified that PIDS experience the isolation of teams within R&D as something negative, since it removed incentives for delivering on time as illustrated below:

> *"Unclarity arises when PIDS work toward the customer while R&D continues development at the same time; responsibilities are more difficult to define when development is not done as committed"* (From PIDS)

Clearly, PIDS expect R&D to deliver on time, but actually experience that there is no incentive for R&D to deliver as committed, since there exist several workarounds for incorporation of late features. On the other hand, R&D expects that they can work according to SD with possibility to e.g. deliver partial features without negative effects. However, with PIDS following a plan, friction is experienced when changes are made that affects the plan with respect to time and content of the release.

F6. Friction in communicating delays and understanding their impact

From the result categories R2 and R3, friction has been identified in relation to communication between R&D and PIDS regarding delayed deliveries, tightly related to both friction F3, F4 and F5:

> *"It was not escalated from R&D that the feature was not in the release, it felt as if they did not realize, did not want to realize or did not listen to the team"* (From PIDS)

As illustrated by the quote, PIDS expect R&D to communicate delays or scope cuts, but experience that R&D does not communicate early enough. On the other hand, R&D expects that PIDS do their own thing and are not affected by delays, but PIDS actually has to redo the plan if delays occur. For the purpose of communicating changes and delays, formal ways are present in the SD process. However, friction has still arisen, indicating that the formal ways might not be clear enough or not sufficiently agile in relation to the balanced way of working.

F7. Friction regarding the possible degree of documenting features

From R1 and R2, it is clear that friction exists between R&D and PIDS regarding documentation of features and the expectation to receive documents at a certain point in time during the release cycle. This is illustrated through the following quote:

> *"PIDS are asking for documentation early but it can often not be delivered until the teams are done with developing the feature"* (From R&D)

With the transition to SD, the focus has shifted from traditional development with extensive documentation toward agility where documentation is seen as waste and therefore should be reduced as much as possible. With this agility, R&D has no possibility to deliver documents up-front when little is known or decided. In addition, documents detailing progress in the team exist during feature development but they are usually not delivered to PIDS along with the feature. However, since PIDS are still using a traditional plan-based process, they expect R&D to deliver documents as before the SD transition. For PIDS, these documents are important and beneficial in order to prepare their work. Without them, an obstacle is created hindering PIDS to follow their planned process.

5.3 Validation Survey Findings

From the survey that was conducted through a questionnaire with the 13 interviewees, validation data was gathered in relation to the seven instances of friction detailed in the previous section. For each friction, respondents were asked to rank the degree of impact on their individual work, the degree of presence of the friction before SD was introduced and if so, how the negative impact has changed after the SD introduction. The consolidated result from the questionnaire is shown in Table 4.

Table 4 Validation questionnaire result

ID	Impact on Own Work	Presence Before SD	Change of Impact
F1	Medium	High	→
F2	Medium	High	→
F3	Very high	Medium	↗
F4	High	Medium	↗
F5	High	High	↗
F6	High	Medium	↗
F7	High	Medium	↗

From Table 4 it is clear that all frictions were present before SD was introduced, although to a greater extend between PM and R&D than between R&D and PIDS. In contrast, friction between R&D and PIDS impact the daily work more than friction between PM and R&D. Additionally, the survey indicates that the negative impact has increased between R&D and PIDS while remaining unaffected between PM and R&D by the transition to SD. Together, this indicates that the relation between R&D and PIDS is to a larger extent exposed to SD related friction than the relation between PM and R&D.

Furthermore, respondents distributed a total of 100 points between the seven frictions, assigning most points to the friction they felt most urgent to minimize in order to achieve higher speed end-to-end in the organization. The average distribution from the 100 points ranking is illustrated in Figure 7.

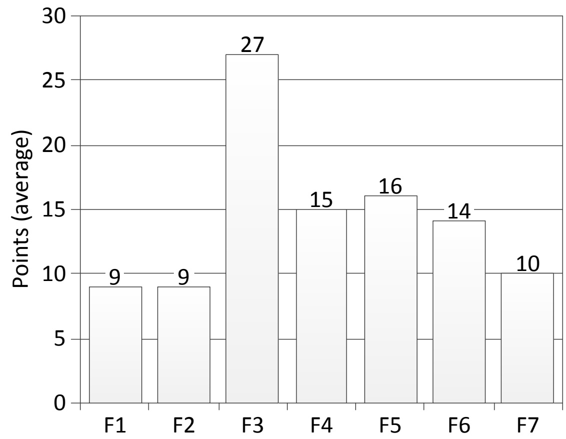

Fig. 7 Average distribution of respondents' priority ranking

From Figure 7 it is clear that the prioritization made by respondents correlates to the results of Table 4, i.e. F3 is the friction that affect the interviewees the most as well as the friction most urgent to resolve.

Apart from F3, it is clear that respondents ranked friction between R&D and PIDS as most urgent to resolve. This indicates that the transition to SD has impacted the relation between R&D and PIDS more than the relation between PM and R&D. In that context, F3 is hard to take into account since it appears between PM and R&D but has effects further down the line between R&D and PIDS. With this cross-organizational effect, the reason for F3 being ranked as more urgent might be explained by the simple notation that more employees are affected by it, i.e. it has a higher end-to-end impact.

6 Discussion

The result from this case study shows that the end-to-end perspective, i.e. the focus on optimizing the whole rather than separate organizational units, can be improved throughout the studied unit. This is most apparent in the friction regarding understanding others' way of working, where it was identified that employees within both R&D and PIDS were reluctant to gain deeper knowledge of each others' processes. It can be argued that without this knowledge, it becomes more difficult for the units to optimize their work in a way that is beneficial for the entire organization, i.e. the units risk sub-optimization.

To concretize, interview findings indicate that PM and R&D have embraced mechanisms of SD that allow for changes in scope and reprioritization late in the development cycle, seen by PIDS as removed incentives for delivering on time and a lack of communicating delays and understanding their impact. Even though PIDS experience extra work load with scope changes near the end of a development cycle, SD has enabled a significant reduction of feature throughput time. Thus, it can be argued that PIDS are not considering the whole organization and its opportunity to satisfy customer requests, i.e. a mutual understanding of beneficial scope changes exists between PM and R&D, but its positive effects from an end-to-end perspective are not acknowledged by PIDS.

This is further supported by the friction regarding the possible degree of documenting features, where PIDS requests early delivery of documents from R&D, but does not realize that R&D cannot produce these documents upfront. Although, to ensure agility in development with high throughput, R&D are forced to postpone documentation of features and releases. While affecting PIDS negatively, this can be seen as a prerequisite for an end-to-end optimization.

In contrast, a similar lack of understanding was not identified between PM and R&D, but as the validation survey indicated, friction between the two units were to a less extent affected by the SD introduction. For instance, this might be the result of a matured relationship between PM and R&D. However, it might also be an indication that PM has become more agile as an adaption to changing market conditions, improving their capability of understanding R&D's agile practices introduced with SD.

Despite the reason, friction identified between PM and R&D seems to concern priorities in the value-stream of a feature, i.e. where to focus the organization's resources in order to optimize feature throughput from an end-to-end perspective. For instance, priorities differ regarding the responsibility to detail customer requirements, where R&D expects PM to mediate details from customers but PM fears loosing customer focus. It can therefore be argued that PM puts most value on gathering and adapting customer requirements in contrast to R&D who values understanding details of features currently in development the most.

As a possible consequence of this valuation conflict, friction in the degree of customer involvement has arisen. With R&D putting high value on acquiring feature details in combination with PM putting low value on acting as surrogated customer while discouraging direct customer involvement, it can be argued that further improvements from a end-to-end perspective is difficult to achieve unless the units are willing to adapt or change their current way of working.

Altogether, the lack of understanding between R&D and PIDS and the difference in valuation between PM and R&D indicates a need for expansion of the agile practices currently present in R&D to both PM and PIDS in order to instigate an end-to-end perspective.

These findings are supported by several previous studies. First, Karlström and Runeson concluded in their research that cross-organizational acceptance is crucial for a successful combination of agile and plan-based processes [12, 13]. This supports the above discussion where it was argued that insufficient understanding of others' way of working inhibits optimization of the whole, i.e. without a cross-organizational end-to-end perspective the combination of agility and planning is less likely to succeed.

Second, the research by Petersen and Wohlin indicates that transitioning from plan-based to agile software development in a large-scale setting increases product packaging and release effort due to increased number of releases [20]. Similar observations have been made during this study, where the effect of enabling faster feature throughput in R&D has resulted in increased release effort for PIDS. However, it is here distinguished that the increased release effort is beneficial from an end-to-end perspective.

Third, Petersen and Wohlin suggested in another study that the release organization should be integrated with the development organization [21]. This suggestion can directly be related with this study, strengthening the observation that PIDS's separation from R&D has negative consequences on optimization of the whole organization. Furthermore, it provides a concrete improvement suggestion, which is in line with the above discussion on expanding agile practices from R&D to PIDS.

Finally, Lindvall et al. concluded in their research that clarifying and specifying the interfaces between agile and plan-based parts is required for a successful balance [15]. This conclusion supports the finding in this study that indicates how a lack of clear interfaces between organizational units decreases the possibility to improve work in a way that is beneficial for the whole organization, i.e. obtaining a successful balance.

Despite the support for this study's findings in previous research and related work, it should not be ruled out that the findings might have other causes. For instance, it is important to notice that the study was conducted at a single development unit with a condensed selection of interviewees. Although selected to cover relevant roles as well as knowledge of the previous way of working, a tendency amongst interviewees were to relate to the ideal theoretical and not the actual way of working when making comparisons to the current state of SD.

Furthermore, it should be noted that through the use of grounded theory, the extraction and grouping of statements from interview transcripts is subjective to the authors' interpretations. Additionally, some of the derived friction instances could possibly be combined to a single entity, e.g. F3, F5 and F6 could be joined but were kept separate in order to visualize their cross-organizational presence.

For further research, it is specifically suggested to deeper investigate how organizational structures incorporating both agility and planning affect the end-to-end perspective. Furthermore, the evolution of SD at the studied Ericsson unit indicates a need for research focusing on identifying enablers to partially or completely transition a large-scale software development organization to an agile way of working.

For the studied unit in particular, it is suggested to incorporate PM and PIDS in the agile way of working within R&D in order to reduce the observed friction. With more incorporated organizational units, the understanding of others' work and valuations should increase, resulting in lower barriers to achieve a cross-organizational end-to-end perspective.

For instance, PM can become more agile through co-location with managers of R&D and PIDS. That would enable them to become more flexible in their communication, which due to higher continuity could trigger a consciousness of PM to consider the whole organization when a promise is made or a decision is taken.

In addition, PIDS can become more agile through inclusion in the cross-functional reams in R&D. This could be achieved through creation of linking roles, i.e. dedicating a PIDS employee part-time in a team-role having an outspoken focus on the release activities. Thus, it would enable instant information from PIDS to other team members within R&D while providing PIDS with better insight into ongoing R&D activities, altogether acting to shorten feedback loops.

However, barriers exist to spread agility in the organization. For example, PM must accept to dedicate more of their time to development and release rather than gathering and adapting customer requirements. Additionally, PIDS would need more resources in order to have fixed roles in R&D activities and their commitment to customer time-slots inhibits them from becoming fully agile in the current market environment.

7 Conclusion

The intention of this study was to characterize friction between agility and planning in a large-scale software development setting, while identifying the effect of friction on the end-to-end speed in the organization. Through a simple friction framework capturing employees' expectations and actual observations, it was identified that friction was present between agile and plan-based parts of the organization.

The friction identified was related to responsibilities in detailing customer requirements, use of in-house surrogated customers, committing to a scope or allowing development to take the time needed, understanding others' way of working, incitements for delivering on time and the need for communicating delays.

Altogether, the findings support that the end-to-end perspective can be improved throughout the organization. With the current separation of agile and plan-based organizational units, a clear lack of understanding others' work and valuations limits further optimizations of the whole organization. It is therefore suggested to incorporate plan-based units into the agile way of working to lower barriers of achieving a cross-organizational end-to-end perspective.

These findings are important contributions to the study of large-scale agile software development thanks to its contribution of empirical data on a current sparsely investigated subject. In terms of future research, it is suggested to deeper investigate end-to-end perspective in relation to organizational structure and to identify enablers for large-scale agile transitions.

References

1. Agile Alliance: Agile Manifesto. http://www.agilemanifesto.org [Last accessed 110217] (2001)
2. Åkerman, N.: The necessity of friction: nineteen essays on a vital force. Physica-Verlag (1993)
3. Beck, K., Andres, C.: Extreme Programming Explained: Embrace Change (2nd Edition). Addison-Wesley Professional (2004)
4. Boehm, B., Turner, R.: Balancing agility and discipline: A guide for the perplexed. Addison-Wesley Professional (2003)
5. Boehm, B., Turner, R.: Using risk to balance agile and plan-driven methods. Computer **36**(6), 57–66 (2003)
6. Cao, L., Mohan, K., Xu, P., Ramesh, B.: How extreme does extreme programming have to be? adapting xp practices to large-scale projects. Hawaii International Conference on System Sciences **3**, 30,083c (2004)
7. Cao, L., Ramesh, B.: Agile requirements engineering practices: An empirical study. IEEE Software **25**(1), 60–67 (2008)
8. Cockburn, A.: Agile Software Development. Addison-Wesley (2002)
9. Däcker, B.: Concurrent functional programming for telecommunications: A case study of technology introduction. Ph.D. thesis, KTH Royal Institute of Technology Stockholm (2000)
10. Dybå, T., Dingsøyr, T.: Empirical studies of agile software development: A systematic review. Information and Software Technology **50**(9-10), 833–859 (2008)
11. Highsmith, J., Cockburn, A.: Agile software development: the business of innovation. Computer **34**(9), 120–127 (2002)
12. Karlström, D., Runeson, P.: Combining agile methods with stage-gate project management. IEEE Software **22**(3), 43–49 (2005)
13. Karlström, D., Runeson, P.: Integrating agile software development into stage-gate managed product development. Empirical Software Engineering **11**(2), 203–225 (2006)
14. Liker, J.K.: The Toyota way: 14 management principles from the world's greatest manufacturer. McGraw-Hill Professional (2004)
15. Lindvall, M., Muthig, D., Dagnino, A., Wallin, C., Stupperich, M., Kiefer, D., May, J., Kähkönen, T.: Agile software development in large organizations. Computer **37**, 26–34 (2004)
16. Mehta, M., Anderson, D., Raffo, D.: Providing value to customers in software development through lean principles. Software Process: Improvement and Practice **13**(1), 101–109 (2008)
17. Middleton, P.: Lean software development: Two case studies. Software Quality Journal **9**(4), 241–252 (2001)
18. Middleton, P., Flaxel, A., Cookson, A.: Lean software management case study: Timberline inc. In: Baumeister, H., Marchesi, M., Holcombe, M. (eds.) Extreme Programming and Agile Processes in Software Engineering, *Lecture Notes in Computer Science*, vol. 3556, pp. 1297–1298. Springer Berlin / Heidelberg (2005)
19. Petersen, K.: Implementing Lean and Agile Software Development in Industry. Blekinge Institute of Technology (2010)
20. Petersen, K., Wohlin, C.: A comparison of issues and advantages in agile and incremental development between state of the art and an industrial case. Journal of Systems and Software **82**(9), 1479 – 1490 (2009)
21. Petersen, K., Wohlin, C.: The effect of moving from a plan-driven to an incremental software development approach with agile practices. Empirical Software Engineering **15**(6), 654–693 (2010)
22. Poppendieck, M., Poppendieck, T.: Lean Software Development: An Agile Toolkit. Addison-Wesley Longman Publishing Co., Inc. (2003)
23. Runeson, P., Höst, M.: Guidelines for conducting and reporting case study research in software engineering. Empirical Software Engineering **14**(2), 131–164 (2009)
24. Strauss, A.L., Corbin, J.M.: Basics of qualitative research: Techniques and procedures for developing grounded theory. Sage Publications, Inc (1998)
25. Sutherland, J.: Agile can scale: Inventing and reinventing scrum in five companies. Cutter IT Journal **14**(12), 5–11 (2001)
26. Talby, D., Hazzan, O., Dubinsky, Y., Keren, A.: Agile software testing in a large-scale project. IEEE Software **23**, 30–37 (2006)
27. Tomaszewski, P., Berander, P., Damm, L.O.: From traditional to streamline development - opportunities and challenges. Software Process: Improvement and Practice **13**(2), 195–212 (2008)
28. Womack, J.P., Jones, D.T., Roos, D.: The Machine That Changed the World : The Story of Lean Production. Simon & Schuster New York (2007)

Appendix I – Interview Questions

QID	Question/Information
A	Non-disclosure of interviewees opinions, how the recording and findings will be used
B	Describe the background and purpose of the study
C	Set the focus on agility and planning
D	Describe common terms that will be used during the interview
Q1	Formal role and performed activities – Can you briefly describe your history at Ericsson? – What is your current role and what responsibilities do you have? – What actual activities do you perform in your work? – Do you perform anything outside your role or responsibility?
Q2	Observed friction in general – Given the vision of delivering to any customer at any time, what friction can you see between R&D and PIDS (Product Integration, Deployment and Support)?
Q3	Friction in relation to the way of working – Would you say that your current way of working is agile, plan-based or a mix? How does this affect your daily work? – Has the vision of delivering to any customer at any time affected your own work and interaction to other departments? – Do you feel that your ability to control your own (and if applicable also others') work has changed after SD (Streamline Development) was introduced?
Q4	Friction in relation to organizational structure – Given that SD deliberately has separated R&D and PIDS, have this created any problems for you or other colleagues/stakeholders? – Do you feel that responsibilities regarding product responsibility, network impact reports and upgrades is clear after delivery from R&D to PIDS (i.e. after PRA)? – Do you feel that your communication with customers is sufficient? Do you get feedback from customers regularly? – Where needed, is the communication interface between R&D and PIDS (or between other agile/plan-based parts) clear and specified anywhere? – Do you know your colleagues' skills and knowledge areas? Do you know what they are working on right now? – During your time at Ericsson, have you been included in work at other parts of the organization?
Q5	Friction in relation to technology and external forces – How do you think that the vision of delivering to any customer at any time involves the customer? – Does interoperability to other network products, including competitors', require a limitation of our flexibility? Should we develop to support everything requested from customers?

	– How do you think FOA (First Office Application) will be driven going forward? Does the current vision impose any changes to the current procedure? – Do you have tools capable enough to support your work according to the vision of delivering to any customer at any time?
Q6	Prioritizing friction and improvements – Given the friction we discussed, what impacts your work the most? What are your top three observed frictions? – Do you have any suggestions on how the friction we discussed can be minimized or removed? – What do you feel is working really well in your current way of working? Is this related to the vision in any way?
Q7	Open-ended questions – Is there anything you expected us to ask that we did not? – Do you have any questions for us so far?

Appendix II – Survey Questions

Survey for validation and prioritization of friction between agile software development and plan-based business processes in the different organizational parts

F1: Friction in the responsibilities to detail customer requirements						
A – To what degree was this friction present prior to the introduction of Streamline Development?						
Small	**-2**	**-1**	**0**	**1**	**2**	**Large**
B – How has the friction's (negative) impact changed after the introduction of Streamline Development?						
Smaller	**-2**	**-1**	**0**	**1**	**2**	**Larger**
C – To what degree does the friction impact your individual work?						
Small	**-2**	**-1**	**0**	**1**	**2**	**Large**

F2: Friction in the degree of customer involvement						
A – To what degree was this friction present prior to the introduction of Streamline Development?						
Small	**-2**	**-1**	**0**	**1**	**2**	**Large**
B – How has the friction's (negative) impact changed after the introduction of Streamline Development?						
Smaller	**-2**	**-1**	**0**	**1**	**2**	**Larger**
C – To what degree does the friction impact your individual work?						
Small	**-2**	**-1**	**0**	**1**	**2**	**Large**

F3: Friction in committing to a scope or allowing the time needed by teams						
A – To what degree was this friction present prior to the introduction of Streamline Development?						
Small	**-2**	**-1**	**0**	**1**	**2**	**Large**
B – How has the friction's (negative) impact changed after the introduction of Streamline Development?						
Smaller	**-2**	**-1**	**0**	**1**	**2**	**Larger**
C – To what degree does the friction impact your individual work?						
Small	**-2**	**-1**	**0**	**1**	**2**	**Large**

F4: Friction in understanding others' way of working						
A – To what degree was this friction present prior to the introduction of Streamline Development?						
Small	**-2**	**-1**	**0**	**1**	**2**	**Large**
B – How has the friction's (negative) impact changed after the introduction of Streamline Development?						
Smaller	**-2**	**-1**	**0**	**1**	**2**	**Larger**
C – To what degree does the friction impact your individual work?						
Small	**-2**	**-1**	**0**	**1**	**2**	**Large**

F5: Friction in incentives for delivering on time						
A – To what degree was this friction present prior to the introduction of Streamline Development?						
Small	**-2**	**-1**	**0**	**1**	**2**	**Large**
B – How has the friction's (negative) impact changed after the introduction of Streamline Development?						
Smaller	**-2**	**-1**	**0**	**1**	**2**	**Larger**
C – To what degree does the friction impact your individual work?						
Small	**-2**	**-1**	**0**	**1**	**2**	**Large**

F6: Friction in communicating delays and understanding their impact						
A – To what degree was this friction present prior to the introduction of Streamline Development?						
Small	**-2**	**-1**	**0**	**1**	**2**	**Large**
B – How has the friction's (negative) impact changed after the introduction of Streamline Development?						
Smaller	**-2**	**-1**	**0**	**1**	**2**	**Larger**
C – To what degree does the friction impact your individual work?						
Small	**-2**	**-1**	**0**	**1**	**2**	**Large**

F7: Friction regarding the possible degree of documenting features						
A – To what degree was this friction present prior to the introduction of Streamline Development?						
Small	**-2**	**-1**	**0**	**1**	**2**	**Large**
B – How has the friction's (negative) impact changed after the introduction of Streamline Development?						
Smaller	**-2**	**-1**	**0**	**1**	**2**	**Larger**
C – To what degree does the friction impact your individual work?						
Small	**-2**	**-1**	**0**	**1**	**2**	**Large**

Prioritization: Divide totally 100 points between frictions F1-F7 below; give each friction 0-100 points where each point indicates which friction you subjectively find **most urgent to resolve in order to receive a better flow in the organization.**

	Number of points (0-100 per friction, 100 in total)
F1	
F2	
F3	
F4	
F5	
F6	
F7	

Appendix III – Example Material from Workshop